

BBBBBBBBBBBB		000000000		000000000		TTTTTTTTTTTT		SSSSSSSSSS
BBBBBBBBBBBB		000000000		000000000		TTTTTTTTTTTT		SSSSSSSSSS
BBBBBBBBBBBB		000000000		000000000		TTTTTTTTTTTT		SSSSSSSSSS
BBB	BBB	000	000	000	000	TTT	SSS	
BBB	BBB	000	000	000	000	TTT	SSS	
BBB	BBB	000	000	000	000	TTT	SSS	
BBB	BBB	000	000	000	000	TTT	SSS	
BBB	BBB	000	000	000	000	TTT	SSS	
BBB	BBB	000	000	000	000	TTT	SSS	
BBB	BBB	000	000	000	000	TTT	SSS	
BBBBBBBBBBBB		000	000	000	000	TTT	SSS	
BBBBBBBBBBBB		000	000	000	000	TTT	SSS	
BBBBBBBBBBBB		000	000	000	000	TTT	SSS	
BBB	BBB	000	000	000	000	TTT	SSS	
BBB	BBB	000	000	000	000	TTT	SSS	
BBB	BBB	000	000	000	000	TTT	SSS	
BBB	BBB	000	000	000	000	TTT	SSS	
BBB	BBB	000	000	000	000	TTT	SSS	
BBB	BBB	000	000	000	000	TTT	SSS	
BBBBBBBBBBBB		000000000		000000000		TTT	SSSSSSSSSS	
BBBBBBBBBBBB		000000000		000000000		TTT	SSSSSSSSSS	
BBBBBBBBBBBB		000000000		000000000		TTT	SSSSSSSSSS	

```
IIIIII  NN  NN  IIIIII  TTTTTTTTTT  PPPPPPPP  GGGGGGGG  FFFFFFFF  IIIIII  LL
IIIIII  NN  NN  IIIIII  TTTTTTTTTT  PPPPPPPP  GGGGGGGG  FFFFFFFF  IIIIII  LL
  II  NN  NN  II  TT  PP  GG  FF  II  LL
  II  NN  NN  II  TT  PP  GG  FF  II  LL
  II  NNNN  NN  II  TT  PP  GG  FF  II  LL
  II  NNNN  NN  II  TT  PP  GG  FF  II  LL
  II  NN  NN  II  TT  PP  GG  FF  II  LL
  II  NN  NN  II  TT  PP  GG  FF  II  LL
  II  NN  NN  II  TT  PP  GG  FF  II  LL
  II  NN  NN  II  TT  PP  GG  FF  II  LL
  II  NN  NN  II  TT  PP  GG  FF  II  LL
IIIIII  NN  NN  IIIIII  TT  PP  GG  FF  IIIIII  LL
IIIIII  NN  NN  IIIIII  TT  PP  GG  FF  IIIIII  LL
LL
```

```
LL  IIIIII  SSSSSSSS
LL  IIIIII  SSSSSSSS
LL  II  SS
LL  II  SS
LL  II  SS
LL  II  SS
LL  II  SSSSSS
LL  II  SSSSSS
LL  II  SS
LL  II  SS
LL  II  SS
LL  II  SS
LLLLLLLLLL  IIIIII  SSSSSSSS
LLLLLLLLLL  IIIIII  SSSSSSSS
```


INITPGFIL
Table of contents

J 11
- Initialize a Page File Control Block 15-SEP-1984 23:53:03 VAX/VMS Macro V04-00

Page 0

(1)	55	DECLARATIONS
(1)	101	INSTALL PAGE OR SWAP FILE
(1)	356	FIND PFL SLOT Find free slot in PFL vector
(1)	424	CHECK ARG LIST
(1)	469	FIND MAXVBN Calculate modified MAXVBN parameter
(1)	504	INIT_BITMAP

```
0000 1 .TITLE INITPGFIL - Initialize a Page File Control Block
0000 2 .IDENT 'V04-000'
0000 3 :
0000 4 :*****
0000 5 :
0000 6 :* COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 7 :* DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 8 :* ALL RIGHTS RESERVED.
0000 9 :
0000 10 :* THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 11 :* ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 12 :* INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 13 :* COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 14 :* OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 15 :* TRANSFERRED.
0000 16 :
0000 17 :* THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 18 :* AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 19 :* CORPORATION.
0000 20 :
0000 21 :* DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 22 :* SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 23 :
0000 24 :
0000 25 :*****
0000 26 :
0000 27 :
0000 28 :++
0000 29 : Facility: SYSGEN Utility
0000 30 :
0000 31 : Abstract: This module isolates the procedure to initialize the
0000 32 : secondary page file control blocks. The procedure was
0000 33 : previously located in module RMSCONIO.
0000 34 :
0000 35 : Environment: The code in this procedure executes in kernel mode.
0000 36 :
0000 37 : Author: R.I. Hustvedt, Creation Date: 7-Sep-1977
0000 38 :
0000 39 : Modified by:
0000 40 :
0000 41 : V03-009 MSH0001 Maryann Hinden 27-Jun-1983
0000 42 : Fix truncation error.
0000 43 :
0000 44 : V03-008 BLS0223 Benn Schreiber 13-May-1983
0000 45 : Fix truncation errors
0000 46 :
0000 47 : V03-007 WMC0001 Wayne Cardoza 31-Jul-1982
0000 48 : Add flag to prevent setting of PFL$M_INITED.
0000 49 :
0000 50 : V03-006 KDM0002 Kathleen D. Morse 28-Jun-1982
0000 51 : Added $PRDEF.
0000 52 :
0000 53 :--
0000 54 :
0000 55 : .SUBTITLE DECLARATIONS
0000 56 :
0000 57 :
```



```
0000 58 : INCLUDE FILES:
0000 59 :
0000 60 :
0000 61 $DYNDEF : Dynamic structure identification codes
0000 62 $IPLDEF : Symbolic IPL codes
0000 63 $PFLDEF : Page file control block
0000 64 $PRDEF : Processor register numbers
0000 65 $PTRDEF : Pointer control block
0000 66 $PTEDEF : Page table entry layout
0000 67 $RSNDEF : Resource codes
0000 68 $SSDEF : System status codes
0000 69 $SYSGMSGDEF : SYSGEN message definitions
0000 70 $WCBDEF : Window control block
0000 71 :
0000 72 :
0000 73 : EQUATED SYMBOLS:
0000 74 :
0000 75 :
0000 76 : Offsets from AP
0000 77 :
00000004 0000 78 FILESIZE = 4 : Size of page or swap file
00000008 0000 79 WCBADDR = 8 : Address of WCB that maps file
0000000C 0000 80 : Caution - the next two parameters fit in a single word.
0000000F 0000 81 MAXVBN = 12 : Largest VBN in file that can be used (24 b
00000010 0000 82 FLAGS = 15 : Input flags
00000010 0000 83 PAGEFIDX = 16 : Address in which to return new
00000014 0000 84 : page file index
00000018 0000 85 MINVBN = 20 : Number of blocks not in bitmap
00000018 0000 86 STARTVBN = 24 : Number of blocks marked as "in use"
00000018 0000 87 :
00000018 0000 88 : Offsets from FP
00000018 0000 89 :
FFFFFFFFC 0000 90 PAGE OR SWAP = -4 : 0 => swap file and 1 => page file
FFFFFFFF8 0000 91 PFLVEC_RILIM = -8 : Upper limit for PFL vector search
FFFFFFFF4 0000 92 PFLVEC_LOLIM = -12 : Lower limit for PFL vector search
FFFFFFFF0 0000 93 PFL_L_STARTVBN = -16 : Saved value of STARTVBN(AP)
FFFFFFFEC 0000 94 PFL_L_MINVBN = -20 : Saved value of MINVBN(AP)
00000000 0000 95 :
00000000 0000 96 : Mask for WCB access field
00000000 0000 97 :
00000060 0000 98 WCB_MASK = WCB$M_COMPLETE ! WCB$M_CATHEDRAL
00000060 0000 99
```

```
0000 101 .SBTTL INSTALL PAGE OR SWAP FILE
0000 102
0000 103 :++
0000 104 : Functional Description:
0000 105 :
0000 106 : BOO$INITPAGFIL
0000 107 : BOO$INITSWPFIL
0000 108 :
0000 109 : BOO$INITxxxFIL initializes a page file control block for a page file
0000 110 : or swap file that has just been opened. A bitmap is allocated from
0000 111 : nonpaged pool and set up to indicate that the entire file is
0000 112 : available for use (bitmap is filled with ones). (If the STARTVBN
0000 113 : parameter is specified and nonzero, the first STARTVBN blocks are
0000 114 : initially marked as in use.) The address of the WCB is stored in the
0000 115 : page file control block. If the caller requests it, the index of
0000 116 : this file (used to locate the PFL in the page file control block
0000 117 : vector) can be returned to the caller.
0000 118 :
0000 119 : Input Parameters:
0000 120 :
0000 121 : FILESIZE(AP) Size (in blocks) of the file
0000 122 : WCBADDR(AP) Address of WCB that maps the file
0000 123 : MAXVBN(AP) Parameter that controls largest VBN that may be used (24 bit
0000 124 : FLAGS(AP) Byte of input flags
0000 125 : bit 0 -> do not set PFL$M_INITED
0000 126 :
0000 127 : If the MAXVBN is zero or is larger than ^X003FFFFFF, then the
0000 128 : MAXVBN field in the page file control block is set to
0000 129 : ^X003FFFFFF. Otherwise, PFL$M_MAXVBN is set to the smallest
0000 130 : power of 2 larger than the MAXVBN input parameter.
0000 131 : (See routine FIND_MAXVBN for details.)
0000 132 :
0000 133 : Optional Input Parameters:
0000 134 :
0000 135 : Both of these parameters must be present or both assume the
0000 136 : default values of zero.
0000 137 :
0000 138 : MINVBN(AP) Number of blocks at the start of the file that
0000 139 : are not represented in the bitmap.
0000 140 : (Defaults to zero if not present)
0000 141 : STARTVBN(AP) Number of bits at the start of the bitmap that
0000 142 : are cleared, indicating that the first STARTVBN
0000 143 : blocks are not available for use.
0000 144 : (Defaults to zero if not present)
0000 145 :
0000 146 : Note that the total number of blocks initially available is
0000 147 :
0000 148 : AVAILABLE = FILESIZE - MINVBN - STARTVBN
0000 149 :
0000 150 : Implicit Input:
0000 151 :
0000 152 : MMG$GL_PAGSWPVC Contains the address of vector that locates each
0000 153 : swap file table entry and page file control block
0000 154 :
0000 155 : SGN$GW_PAGFILCT Maximum number of paging files allowed in this
0000 156 : configuration
0000 157 :
```



```
0000 158 : SGNSGW_SWPFILCT Maximum number of swapping files allowed in this
0000 159 : configuration
0000 160 :
0000 161 : IPL is assumed to be zero on entry to these procedures.
0000 162 :
0000 163 : Output Parameters:
0000 164 :
0000 165 : PAGEFIDX(AP) Address in which to return new page file index
0000 166 :
0000 167 : Implicit Output:
0000 168 :
0000 169 : A page file control block and its associated bitmap are allocated
0000 170 : from nonpaged pool. Various fields in the PFL are filled in
0000 171 : according to the input parameters. All bits in the bitmap are set,
0000 172 : indicating an empty file (unless STARTVBN is specified and nonzero,
0000 173 : in which case, the first STARTVBN bits are cleared, indicating that
0000 174 : the associated blocks are initially in use.) The address of the map
0000 175 : is stored in the page file control block. Finally, the page file
0000 176 : control block address is stored in the first empty entry in the page
0000 177 : file control block vector.
0000 178 :
0000 179 : MMG$GL_MAXPFIDX This cell contains the index of the most recently
0000 180 : installed paging file. If entry is at BOO$INITPAGFIL,
0000 181 : then this cell is updated.
0000 182 :
0000 183 : Completion Status:
0000 184 :
0000 185 : R0 low bit set indicates success.
0000 186 :
0000 187 : R0 low bit clear indicates error.
0000 188 :
0000 189 : SSS_INSFMEM Insufficient nonpaged pool for bitmap
0000 190 :
0000 191 : SYSG$_SWAPAGINS There is no more room in the page file
0000 192 : control block vector. The number of page or
0000 193 : swap files specified by the appropriate
0000 194 : SYSGEN parameter have already been installed.
0000 195 :
0000 196 : SYSG$_EMPTYFILE Page or swap files of zero length cannot be
0000 197 : installed.
0000 198 :
0000 199 : SSS_PARTMAPPED File does not have all of its mapping pointers
0000 200 : permanently resident and there is not enough
0000 201 : nonpaged pool to allocate an extended window
0000 202 : control block.
0000 203 :--
0000 204 :
00000000 205 : .PSECT PAGED_CODE RD,NOWRT,EXE, LONG
0000 206 :
0000 207 : .ENABLE LOCAL_BLOCK
0000 208 :
0000 209 : BOO$INITPAGFIL::
0000 210 : .WORD ^M<R2,R3,R4,R5,R6,R7,R8,R9> ; Entry mask
0000 211 : PUSH #1 ; Store code that distinguishes entry
0000 212 : MOVZWL G^<SGNSGW_PAGFILCT-EXESA,SYSPARAM+MMG$A_SYSPARAM>,-(SP)
0000 213 : ; Zero extend page file count
0000 214 : BEQL 10$ ; Count of zero prevents installation
```

01 DD 0002
7E 00000000 GF 3C 0004
10 13 000B 213


```
51 00000000'GF 3C 000D 215      MOVZWL  G^SGN$GW_SWAPFILCT,R1 ; Zero extend swap file count
    6E 51 C0 0014 216      ADDL2   R1,(SP) ; Store sum as upper limit PFL index
    6E D7 0017 217      DECL    (SP) ; Account for zero origin
    51 DD 0019 218      PUSHL   R1 ; Swap file count is lower limit
    1B 11 001B 219      BRB     20$ ; Join common code
    001D 220
50 007C8072 8F D0 001D 221 10$: MOVL   #SYS$G_SWAPAGINS,R0 ; Indicate error status
    04 0024 222      RET ; and return
    0025 223
    0025 224 BOO$INITSWPFIL::
    03FC 0025 225      .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9> ; Entry mask
    DD 0027 226      PUSHL   #0 ; Store entry point code
51 00000000'GF 3C 0029 227      MOVZWL  G^SGN$GW_SWAPFILCT,R1 ; Zero extend swap file count
    7E 51 01 C3 0030 228      SUBL3   #1,R1,-(SP) ; Modified swap file count is upper limit
    E7 13 0034 229      BEQL    10$ ; (Count of one prevents installation)
    01 DD 0036 230      PUSHL   #1 ; and 1 is the lower limit
    0038 231
    0038 232 ; Calculate value for PFL$L_MAXVBN
    0038 233
    5E 08 C2 0038 234 20$: SUBL2   #8,SP ; Allocate space for MINVBN and STARTVBN
    00000083'EF 16 003B 235      JSB    CHECK_ARG_LIST ; Get MINVBN and STARTVBN parameters
56 0C AC 18 00 EF 0041 236      EXTZV  #0,#24,MAXVBN(AP),R6 ; Get MAXVBN parameter
    OE 13 0047 237      BEQL    30$ ; If not present, use default
    003FFFFFF 8F 56 D1 0049 238      CMPL  R6,#PTES$M_PGFLVB ; Also use default if input parameter
    05 1E 0050 239      BGEQU   30$ ; is larger than default value
    00AD 30 0052 240      BSBW    FIND_MAXVBN ; Otherwise, perform a detailed calculation
    07 11 0055 241      BRB     40$ ; ... and continue in line
    0057 242
56 003FFFFFF 8F D0 0057 243 30$: MOVL   #PTES$M_PGFLVB,R6
    005E 244
    005E 245 ; Calculate modified file size
    005E 246
7E 04 AC EC AD C3 005E 247 40$: SUBL3   PFL$L_MINVBN(FP),FILESIZE(AP),-(SP) ; Get actual size
    03 1A 0064 248      BGTRU   45$
    008B 31 0066 249 42$: BRW     70$ ; Error if absurd parameters
57 8E 07 CB 0069 250 45$: BICL3   ^B0111,(SP)+,R7 ; Retrieve modified file size
    F7 13 006D 251      BEQL    42$ ; Error if file has no space in it
    56 57 D1 006F 252      CMPL  R7,R6 ; Make sure that file size is smaller
    04 1F 0072 253      BLSSU   50$ ; than MAXVBN.
57 56 07 CB 0074 254      BICL3   ^B0111,R6,R7 ; If not, minimize file size.
    0078 255
    0078 256 ; All mapping pointers must be permanently resident in order that memory
    0078 257 ; management I/O requests can always complete without ACP intervention.
    0078 258
    59 08 AC D0 0078 259 50$: MOVL   WCBADDR(AP),R9 ; Get WCB address from argument list
50 0B A9 9F 8F 8B 007C 260      BICB3   ^C<WCB_MASK>,WCB$B_ACCESS(R9),R0
    0082 261 ; Check COMPLETE and CATHEDRAL
    50 60 8F 91 0082 262      CMPB   #WCB_MASK,R0 ; bits. Error if both bits are not set
    74 12 0086 263      BNEQ    80$ ; Set error status and return
    0088 264
    0088 265 ; Determine bitmap size in bytes
    0088 266
    58 57 08 C7 0088 267      DIVL3   #8,R7,R8 ; Convert bit count to byte count
51 58 25 C1 008C 268      ADDL3   #<PFL$K_LENGTH+1>,R8,R1 ; Determine size of pool allocation
    0090 269 ; (+1 is for stopper byte in bitmap)
    00000000'GF 16 0090 270      JSB    G^EXES$ALONONPAGED ; Allocate a block of pool
    68 50 E9 0096 271      BLBC    R0,90$ ; Quit if allocation failed
```



```
0099 272
0099 273 : Now load the various fields in the page file control block
0099 274 :
0099 275 : R1 = Size of allocation request
0099 276 : R2 = Address of page file control block
0099 277 : R6 = Modified MAXVBN parameter
0099 278 : R7 = Modified FILESIZE parameter
0099 279 : R8 = Bitmap size in bytes
0099 280 : R9 = Address of window control block that completely maps file
0099 281 :
0099 282 MOVAL PFL$$_BITMAPLOC(R2),PFL$$_BITMAP(R2) ; Store address of bitmap
0099 283 CLRL PFL$$_STARTBYTE(R2) ; Let allocator initialize this field
08 A2 51 58 C3 00A0 284 SUBL3 R8,R1,PFL$$_SIZE(R2) ; Store PFL size (excluding bitmap size)
0A A2 23 9B 00A5 285 MOVZBW #DYN$C PFL,PFL$$_TYPE(R2) ; Store type code and clear PFC field
0C A2 59 D0 00A9 286 MOVL R9,PFL$$_WINDOW(R2) ; Store WCB address into PFL
10 A2 8E D0 00AD 287 MOVL (SP)+,PFL$$_VBN(R2) ; Store the offset VBN field
14 A2 58 D0 00B1 288 MOVL R8,PFL$$_BITMAPSIZE(R2) ; Store bitmap size
18 A2 57 F0 AD C3 00B5 289 SUBL3 PFL$$_STARTVBN(FP),R7,PFL$$_FREPAGECNT(R2) ; Free page count
0099 290 ; Is modified file size - STARTVBN
0099 291 BGTRU 60$ ; Keep going if there are free pages
007C807A 8F DD 00BD 292 PUSHL #SYSG$_EMPTYFILE ; Treat error as "file too small"
00000040 EF 17 00C3 293 JMP L^140$ ; Error if absurd parameters
0099 294
0099 295 60$: MOVL R6,PFL$$_MAXVBN(R2) ; Store MAXVBN parameter
0099 296 CLRW PFL$$_ERRORCNT(R2) ; Clear count of potentially bad blocks
22 A2 00000000 GF 90 00D0 297 MOVVB G^MPW$GW MPWPFC,PFL$$_ALLOCISZ(R2) ; Initialize MPW cluster factor
004A 30 00D8 298 BSBW INIT_BITMAP ; Mark all blocks in file as free
5E 04 C0 00DB 299 ADDL2 #4,SP ; Clear STARTVBN value from stack
23 A2 01 94 00DE 300 CLRB PFL$$_FLAGS(R2)
0F AC 01 93 00E1 301 BITB #1,FLAGS(AP) ; Should it be marked useable
04 12 00E5 302 BNEQ 65$ ; No
23 A2 01 88 00E7 303 BISB #PFL$$_INITED,PFL$$_FLAGS(R2) ; Indicate that file is ready
0099 304 65$:
0099 305
0099 306 : Now locate empty PFL vector slot and store PFL address
0099 307
0099 308 MOVQ (SP)+,R4 ; Load index limits into R4 and R5
0099 309 JMP L^LOCKED_CODE_BEGIN
0099 310
0099 311 : Error returns
0099 312
0099 313 70$: MOVL #SYSG$_EMPTYFILE,R0 ; Zero length files cannot be installed
0099 314 RET ; Return error status
0099 315
0099 316 80$: MOVZWL #SS$_PARTMAPPED,R0 ; Set error status code
0099 317 90$: RET ; and return
0099 318
0099 319 .PSECT NONPAGED_CODE RD,NOWRT,EXE,LONG
0099 320
0099 321 LOCKED_CODE_BEGIN:
0099 322 SETIPL LOCK_IPL ; Do at IPL 7 to prevent simultaneous update
0099 323 BSBW FIND_PFL_SLOT ; Locate empty PFL slot and store
0099 324 BLBC R0,130$
0099 325 TSTL (SP)+ ; Entry at BOO$INITPGFIL?
0099 326 BEQL 110$ ; Branch if not
0099 327 MOVL R3,G^MMG$GL_MAXPFIDX ; Otherwise, update PFL index upper limit
0099 328 MOVL #RSN$_PGFILE,R0 ; Report PAGEFILE resource available only
```

```
00000000'GF 16 001B 329 JSB G^SCH$RAVAIL ; when installing paging file
      50 0A D0 0021 330 110$: MOVL #RSN$ SWPFILE,R0 ; Report SWAPFILE resource available when
00000000'GF 16 0024 331 JSB G^SCH$RAVAIL ; any paging or swap file is installed
      51 10 AC D0 002A 332 SETIPL #0 ; Reenable scheduling
      03 13 002D 333 MOVL PAGEIDX(AP),R1 ; Does the caller want the PFL index?
      61 53 D0 0031 334 BEQL 120$ ; Branch if not
      50 01 3C 0033 335 MOVL R3,(R1) ; Otherwise store the PFL index
      3C 0036 336 120$: MOVZWL #SS$_NORMAL,R0 ; Signal success
      04 0039 337 RET ; and return
      003A 338
      003A 339 ; No free slot is available in the page file control block vector. The
      003A 340 ; PFL and its associated bitmap must be deallocated.
      003A 341 ;
      003A 342 ; (SP) Error status code
      003A 343
007C8072 8F DD 003A 344 130$: PUSHL #SYSG$_SWAPAGINS ; Signal a failure
      50 52 D0 0040 345 140$: MOVL R2,R0 ; Get address of PFL
      51 08 A2 3C 0043 346 MOVZWL PFL$W_SIZE(R2),R1 ; Get size of PFL less bitmap size
      51 14 A2 C0 0047 347 ADDL2 PFL$L_BITMAPSZ(R2),R1 ; Add bitmap size
00000000'GF 16 004B 348 JSB G^EXE$DEANONPGDSIZ ; Deallocate the block
      0051 349 SETIPL #0 ; Reenable scheduling
      50 8E D0 0054 350 MOVL (SP)+,R0 ; Restore error status
      04 0057 351 RET ; Return error status to caller
      0058 352
      0058 353
      0058 354 .DISABLE LOCAL_BLOCK
```



```
0058 356 .SUBTITLE FIND_PFL_SLOT Find free slot in PFL vector
0058 357
0058 358
0058 359 :+ This routine locates the first free slot (one pointing to MMG$GL_NULLPFL)
0058 360 : in either the swap file or page file area of the PFL vector and loads
0058 361 : the PFL address passed as an input parameter into that slot.
0058 362
0058 363 : Input parameters:
0058 364 :
0058 365 : R2 = Address of page file control block to be stored in vector
0058 366 : R4 = Index at which search begins
0058 367 : R5 = Index at which search must end
0058 368
0058 369 : (Note that R4 and R5 are inclusive limits)
0058 370 : (Note also that R4 LSSU R5)
0058 371
0058 372 : Implicit input:
0058 373 :
0058 374 : MMG$GL_PAGSWPVC Pointer to page file control block vector
0058 375
0058 376 : Output parameter:
0058 377 :
0058 378 : R3 = Index into PFL vector into which PFL address is stored.
0058 379
0058 380 : Implicit output:
0058 381 :
0058 382 : The PFL address passed into this routine in R2 is loaded into
0058 383 : the empty vector slot located by this routine.
0058 384
0058 385 : Side effects:
0058 386 :
0058 387 : R1 is destroyed
0058 388
0058 389 : Return status:
0058 390 :
0058 391 : R0 = SS$_NORMAL => successful return
0058 392 :
0058 393 : R0 = SS$_NOSLOT => no empty slots are available
0058 394 :-
0058 395
0058 396 FIND_PFL_SLOT:
0058 397 MOVAL G^MMG$GL_NULLPFL,R0 ; This address indicates an empty slot
50 00000000'GF DE 0058 397 MOVL G^MMG$GL_PAGSWPVC,R1 ; Get PFL vector address
51 00000000'GF DO 005F 398 MOVL R4,R3 ; Get initial index value
53 54 DO 0066 399 10$: CMPL (R1)[R3],R0 ; Is this slot free?
50 6143 D1 0069 400 BFQL 30$ ; Equal implies free
F6 53 0A 13 006D 401 AUBLEQ R5,R3,10$ ; If we drop through this loop, then ...
55 F3 006F 402
50 039C 8F 3C 0073 403 20$: MOVZWL #SS$_NOSLOT,R0 ; There is no slot available
05 0078 404 RSB ; and return error code
0079 405
6143 52 DO 0079 407 30$: MOVL R2,(R1)[R3] ; Store PFL address in empty slot
50 01 3C 007D 408 MOVZWL #SS$_NORMAL,R0 ; Indicate success
05 0080 409 RSB ; and return to caller
0081 410
0081 411 : This method of locking pages down while elevating IPL is used because
0081 412 : this module is used by both SYSINIT and SYSGEN. SYSGEN locks pages
```

INITPGFIL
V04-000

F 12

- Initialize a Page File Control Block 15-SEP-1984 23:53:03 VAX/VMS Macro V04-00 Page 9
FIND_PFL_SLOT Find free slot in PFL vect 4-SEP-1984 23:04:22 [BOOTS.SRC]INITPGFIL.MAR;1 (1)

```
0081 413 : into its working set, making this technique unnecessary (but harmless).
0081 414 : SYSINIT does not lock pages into its working set.
0081 415
0081 416 LOCK_IPL:
0008 0081 417 .WORD IPL$_SYNCH ; Value of synchronization IPL
0083 418
0083 419 LOCKED_CODE_END:
0083 420
0083 421 ASSUME <LOCKED_CODE_END - LOCKED_CODE_BEGIN> LE 512
0083 422
```



```
0083 424 .SUBTITLE CHECK_ARG_LIST
0083 425 :+
0083 426 : Check the argument list for the presence of optional parameters. If the
0083 427 : parameters are present, store their values in local storage. If the
0083 428 : parameters are not specified, store their default values of zero.
0083 429 :
0083 430 : Input parameter:
0083 431 :
0083 432 : (AP) = Number of arguments passed to procedure
0083 433 :
0083 434 : Optional input parameters:
0083 435 :
0083 436 : MINVBN(AP) Number of blocks at the start of the file that
0083 437 : are not represented in the bitmap.
0083 438 : (Defaults to zero if not present)
0083 439 : STARTVBN(AP) Number of bits at the start of the bitmap that
0083 440 : are cleared, indicating that the first STARTVBN
0083 441 : blocks are not available for use.
0083 442 : (Defaults to zero if not present)
0083 443 :
0083 444 : Output parameters:
0083 445 :
0083 446 : PFL_L_MINVBN(FP) Set to value of MINVBN(AP) or zero if
0083 447 : that parameter is not present.
0083 448 : PFL_L_STARTVBN(FP) Set to value of STARTVBN(AP) or zero if
0083 449 : that parameter is not present.
0083 450 :-
0083 451 :
0083 452 CHECK_ARG_LIST:
0083 453 CMPB (AP),#6 ; Check for six or more parameters
0086 454 BLSSU 10$ ; Branch if fewer than six
EC AD 06 6C 91 0088 455 MOVQ MINVBN(AP),PFL_L_MINVBN(FP) ; Store optional parameters
14 AC 06 1F 008D 456 RSB
05 008E 457
008E 458 ; The following assumptions demand that the two optional parameters be
008E 459 ; adjacent in the argument list and in local storage so that they can
008E 460 ; be stored (or zeroed) with a single MOVQ (or CLRQ) instruction.
008E 461
008E 462 ASSUME STARTVBN EQ <MINVBN + 4> ; Offsets from AP
008E 463 ASSUME PFL_L_STARTVBN EQ <PFL_L_MINVBN + 4> ; Offsets from FP
008E 464
EC AD 7C 008E 465 10$: CLRQ PFL_L_MINVBN(FP) ; Parameters default to zero
05 0091 466 RSB
0092 467
```

```
0092 469      .SUBTITLE      FIND_MAXVBN      Calculate modified MAXVBN parameter
0092 470
0092 471      ;+
0092 472      ; This routine calculates the smallest power of two that is larger than
0092 473      ; a given integer and returns a value which is that value minus one.
0092 474      ;
0092 475      ; Input parameter:
0092 476      ;
0092 477      ;      R6 = Integer between 0 and ^X3FFFFFF
0092 478      ;
0092 479      ; Output parameter:
0092 480      ;
0092 481      ;      R6 = Integer of the form 2**N - 1 where
0092 482      ;
0092 483      ;      N-1 LSSU log(x) LEQU N
0092 484      ;
0092 485      ;      log(x) is log base 2 of the input parameter
0092 486      ; -
0092 487
00000102 488      .PSECT PAGED_CODE      RD,NOWRT,EXE, LONG
0102 489
0102 490 FIND_MAXVBN:
52 56 D0 0102 491      MOVL      R6,R2
50 01 D0 0105 492      MOVL      #1,R0
51 01 D0 0108 493      MOVL      #1,R1
52 50 D1 010B 494 10$:      CMPL      R0,R2
51 09 1E 010E 495      BGEQU     20$
51 01 78 0110 496      ASHL      #1,R1,R1
50 51 C8 0114 497      BISL2     R1,R0
F2 11 0117 498      BRB        10$
0119 499
56 50 D0 0119 500 20$:      MOVL      R0,R6
05 011C 501      RSB
011D 502
```



```
011D 504 .SUBTITLE INIT_BITMAP
011D 505 :+
011D 506 : Initialize the page file bitmap. The first STARTVBN bits are
011D 507 : cleared, indicating that the associated blocks are initially
011D 508 : allocated. The remaining bits are set, indicating that the remainder
011D 509 : of the file is available.
011D 510 :
011D 511 : Input parameters:
011D 512 :
011D 513 : R2 = Address of page file control block
011D 514 :
011D 515 : PFL$$_BITMAPSIZE(R2) Size in bytes of bitmap
011D 516 : PFL$$_BITMAP(R2) Address of start of bitmap
011D 517 : PFL$_STARTVBN(FP) Number of blocks to mark as in use
011D 518 :
011D 519 : Side effects:
011D 520 :
011D 521 : All of R0 through R9, with the exception of R2, are destroyed.
011D 522 :-
011D 523 :
011D 524 : Page file allocation code assumes that the first byte in its
011D 525 : bitmap never contains all ones. By placing the flags byte
011D 526 : immediately before the beginning of the bitmap and reserving a flag
011D 527 : bit for all time, this function is accomplished.
011D 528 :
011D 529 : ASSUME PFL$$_BITMAPLOC EQ <PFL$$_FLAGS + 1>
011D 530 :
011D 531 : The following table contains the eight possibilities that can exist
011D 532 : for the boundary byte between the portion of the bitmap that indicates
011D 533 : blocks in use and the portion that indicates free blocks.
011D 534 :
011D 535 : BOUNDARY_BYTE:
FF 011D 536 : .BYTE ^B11111111
FE 011E 537 : .BYTE ^B11111110
FC 011F 538 : .BYTE ^B11111100
F8 0120 539 : .BYTE ^B11111000
F0 0121 540 : .BYTE ^B11110000
E0 0122 541 : .BYTE ^B11100000
C0 0123 542 : .BYTE ^B11000000
80 0124 543 : .BYTE ^B10000000
0125 544 :
0125 545 : INIT_BITMAP:
53 62 D0 0125 546 : MOVL PFL$$_BITMAP(R2),R3 : Start of bitmap to R3
58 F0 AD D0 0128 547 : PUSHL #0 : Initialize top of stack to zero
08 13 012A 548 : MOVL PFL$_STARTVBN(FP),R8 : Get STARTVBN parameter into register
58 6E 58 08 012E 549 : BEQL 10$ : Branch if entire file available
59 D4 0130 550 : CLRL R9 : Clear upper half of R8:R9 quadword
7B 0132 551 : EDIV #8,R8,(SP),R8 : Quotient to top of stack
0137 552 : : Remainder to R8
59 D4 0137 553 : CLRL R9 : Set fill character to 00 (null)
19 10 0139 554 : BSBB 20$ : Clear out first half of bitmap
83 DE AF 48 90 013B 555 10$: : MOVBL BOUNDARY_BYTE[R8],(R3)+ : Set/clear boundary byte
50 8E 01 01 0140 556 : ADDL3 #1,(SP)+,R0 : R0 contains number of bytes completed
7E 14 A2 50 C3 0144 557 : SUBL3 R0,PFL$$_BITMAPSIZE(R2),-(SP) : Bytes remaining to top of stack
59 00 D2 0149 558 : MCOML #0,R9 : Set fill character to FF (all ones)
06 10 014C 559 : BSBB 20$ : Set rest of bitmap to all ones
63 94 014E 560 : CLRB (R3) : Set stopper byte at end of bitmap
```

```
SE 04 C0 0150 561 ADDL2 #4,SP ; Reset stack pointer
      05 0153 562 RSB ; and return
      0154 563
      0154 564 :-
      0154 565 ; Set or clear a number of bits in a bitmap
      0154 566
      0154 567 Input parameters:
      0154 568
      0154 569 R3 Bitmap address (updated by this routine)
      0154 570 R9 Fill character (either 00 or FF)
      0154 571 0(SP) Return PC
      0154 572 4(SP) Number of bytes to set or clear
      0154 573
      0154 574 Side effects:
      0154 575
      0154 576 R2 preserved
      0154 577 R3 updated to point one byte beyond bitmap
      0154 578 The rest of R0 through R7 are destroyed
      0154 579 :-
      0154 580
      56 52 DD 0154 581 20$: PUSHL R2 ; Only register worth saving
      57 08 AE 3C 0156 582 MOVZWL 8(SP),R6 ; Low order word of bitmap size to R6
      0A AE 3C 015A 583 MOVZWL 10(SP),R7 ; High order word of bitmap size to R7
      0E 13 015E 584 BEQL 40$ ; Skip loop if one MOVCS will suffice
      0160 585
      63 FFFF 8F 59 63 00 2C 0160 586 30$: MOVCS #0,(R3),R9,#^XFFFF,(R3) ; Initialize (64k - 1) bytes of bitmap
      83 59 90 0168 587 MOVB R9,(R3)+ ; Get the last byte, too
      F2 57 F5 016B 588 SOBGTR R7,30$ ; Go back for next 64k block
      016E 589
      63 56 59 63 00 2C 016E 590 40$: MOVCS #0,(R3),R9,R6,(R3) ; Initialize what's left
      52 8ED0 0174 591 POPL R2 ; Restore PFL address to R2
      05 0177 592 RSB
      0178 593
      0178 594 .END
```


INITPGFIL
Symbol table

- Initialize a Page File Control Block

K 12

15-SEP-1984 23:53:03

VAX/VMS Macro V04-00

Page 14

4-SEP-1984 23:04:22

[BOOTS.SRC]INITPGFIL.MAR;1

(1)

BOOSINITPAGFIL	00000000	RG	02	SYSG\$_SWAPAGINS	= 007C8072
BOOSINITSWPFIL	00000025	RG	02	WCB\$_ACCESS	= 0000000B
BOUNDARY_BYTE	0000011D	R	02	WCB\$_CATHEDRAL	= 00000040
CHECK_ARG_LIST	00000083	R	03	WCB\$_COMPLETE	= 00000020
DYN\$_PFL	= 00000023			WCBADDR	= 00000008
EXESA\$CONONPAGED	*****	X	02	WCB_MASK	= 00000060
EXESA_SYSPARAM	*****	X	02		
EXESDEANONPGDSIZ	*****	X	03		
FILESIZE	= 00000004				
FIND_MAXVBN	00000102	R	02		
FIND_PFL_SLOT	00000058	R	03		
FLAGS	= 0000000F				
INIT_BITMAP	00000125	R	02		
IPL\$_SYNCH	= 00000008				
LOCKED_CODE_BEGIN	00000000	R	03		
LOCKED_CODE_END	00000083	R	03		
LOCK_IPL	00000081	R	03		
MAXVBN	= 0000000C				
MINVBN	= 00000014				
MMG\$_SYSPARAM	*****	X	02		
MMG\$_MAXPFIIDX	*****	X	03		
MMG\$_NULLPFL	*****	X	03		
MMG\$_PAGSWPVC	*****	X	03		
MPW\$_MPWPC	*****	X	02		
PAGEFIDX	= 00000010				
PAGE_OR_SWAP	= FFFFFFFC				
PFL\$_ACLOCSIZ	= 00000022				
PFL\$_FLAGS	= 00000023				
PFL\$_TYPE	= 0000000A				
PFL\$_LENGTH	= 00000024				
PFL\$_BITMAP	= 00000000				
PFL\$_BITMAPLOC	= 00000024				
PFL\$_BITMAPSIZ	= 00000014				
PFL\$_FREPAGECNT	= 00000018				
PFL\$_MAXVBN	= 0000001C				
PFL\$_STARTBYTE	= 00000004				
PFL\$_VBN	= 00000010				
PFL\$_WINDOW	= 0000000C				
PFL\$_INITED	= 00000001				
PFL\$_ERRORCNT	= 00000020				
PFL\$_SIZE	= 00000008				
PFLVEC_HILIM	= FFFFFFFF8				
PFLVEC_LOLIM	= FFFFFFFF4				
PFL_L_MINVBN	= FFFFFFFEC				
PFL_L_STARTVBN	= FFFFFFFF0				
PR\$_IPL	= 00000012				
PTE\$_PGFLVB	= 003FFFFFF				
RSN\$_PGFILE	= 00000004				
RSN\$_SWPFILE	= 0000000A				
SCH\$_RAVAIL	*****	X	03		
SGN\$_PAGFILCT	*****	X	02		
SGN\$_SWPFILCT	*****	X	02		
SS\$_NORMAL	= 00000001				
SS\$_NOSLOT	= 0000039C				
SS\$_PARTMAPPED	= 00000E22				
STARTVBN	= 00000018				
SYSG\$_EMPTYFILE	= 007C807A				

+-----+
! Psect synopsis !
+-----+

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
\$AB\$\$	00000000 (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
PAGED_CODE	00000178 (376.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC LONG
NONPAGED_CODE	00000092 (146.)	03 (3.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC LONG

+-----+
! Performance indicators !
+-----+

Phase	Page faults	CPU Time	Elapsed Time
Initialization	31	00:00:00.08	00:00:00.26
Command processing	125	00:00:00.66	00:00:02.00
Pass 1	299	00:00:09.09	00:00:19.27
Symbol table sort	0	00:00:01.36	00:00:02.78
Pass 2	116	00:00:02.08	00:00:04.31
Symbol table output	9	00:00:00.07	00:00:00.19
Psect synopsis output	2	00:00:00.04	00:00:00.11
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	584	00:00:13.38	00:00:28.92

The working set limit was 1500 pages.
52000 bytes (102 pages) of virtual memory were used to buffer the intermediate code.
There were 50 pages of symbol table space allocated to hold 922 non-local and 26 local symbols.
594 source lines were read in Pass 1, producing 15 object records in Pass 2.
19 pages of virtual memory were used to define 18 macros.

+-----+
! Macro library statistics !
+-----+

Macro library name	Macros defined
_\$255\$DUA28:[BOOTS.OBJ]BOOTS.MLB;1	0
_\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	9
_\$255\$DUA28:[SYSLIB]STARLET.MLB;2	6
TOTALS (all libraries)	15

1004 GETS were required to define 15 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LISS:INITPGFIL/OBJ=OBJ\$:INITPGFIL MSRC\$:INITPGFIL/UPDATE=(ENH\$:INITPGFIL)+EXECML\$/LIB+LIB\$:BOOTS.MLB/LIB

0038 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY